Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

**(ВлГУ)**

**Кафедра информационных систем и программной инженерии**

Лабораторная работа № 2

по дисциплине «Проектирование и

разработка веб-приложений»

Выполнил:

ст. гр. ИРсп-121

Орлова С. В.

Приняла:

Максимова А. С.

Владимир 2023 г.

**Цель работы**

Научится создавать простейшие HTTP сервера с помощью модуля http, получать GET и POST параметры и создавать маршрутизацию.

Научится создавать самоподписанных HTTPS сертификатов, подключать HTTPS к HTTP-серверу NodeJS.

**Задание**

1. Ознакомиться с материалом методического указания, выполните примеры из методички.

2. Создать http-сервер, который решает следующие задачи:

- Работа с GET и POST параметрами

- Маршрутизация

- Переадресация

- Вывод на каждой странице вашего ФИО и группы

3. Создать самоподписываемый сертификат

4. Создать https-сервер, который решает следующие задачи:

- Работа с GET и POST параметрами

- Маршрутизация

- Переадресация

- Вывод на каждой странице вашего ФИО и группы

5. Составить отчет по результатам работы

**Практическая часть**

**Создание базового сервера HTTP**

Код для файла server.js.

Листинг:

var http = require("http");

http.createServer(function(request, response){

console.log("Url: " + request.url);

console.log("Тип запроса: " + request.method);

console.log("User-Agent: " + request.headers["user-agent"]);

console.log("Все заголовки");

console.log(request.headers);

response.end();

}).listen(3000);

Результат работы представлен на рисунке 1.

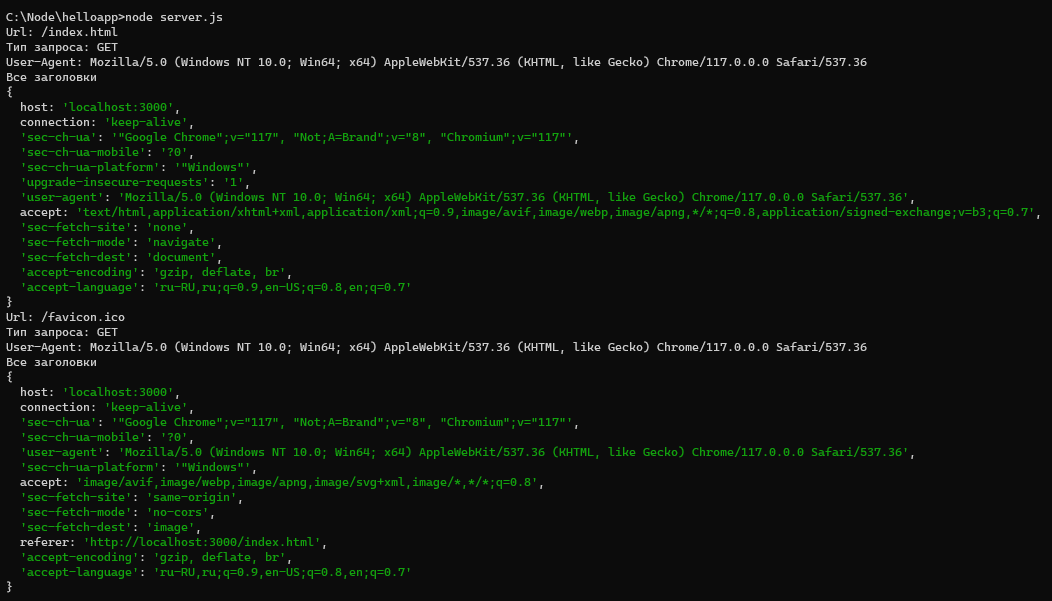


Рисунок 1 – Результат

**Response**

Изменение файла server.js следующим образом.

Листинг:

const http = require("http");

http.createServer(function(request, response){

response.setHeader("UserId", 12);

response.setHeader("Content-Type", "text/html; charset=utf8;");

response.write("<h2>hello world</h2>");

response.end();

}).listen(3000);Результат работы представлен на рисунке 2.

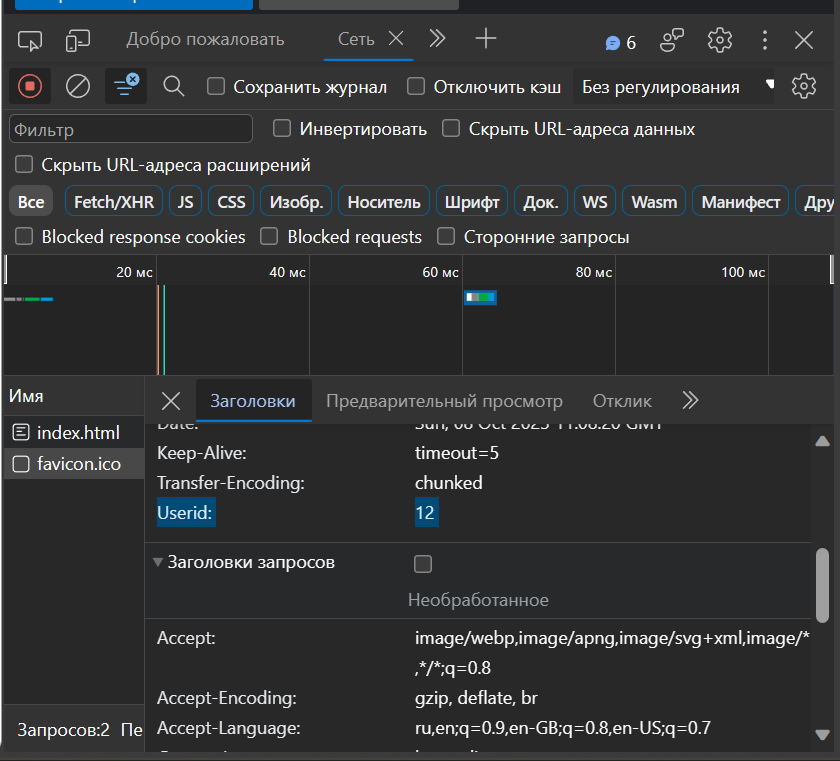


Рисунок 2 – Результат

Листинг:

const http = require("http");

http.createServer(function(request, response){

response.setHeader("Content-Type", "text/html");

response.write("<!DOCTYPE html>");

response.write("<html>");

response.write("<head>");

response.write("<title>Hello Node.js</title>");

response.write("<meta charset=\"utf-8\" />");

response.write("</head>");

response.write("<body><h2>Привет мир!!</h2></body>");

response.write("</html>");

response.end();

}).listen(3000);

Результат работы представлен на рисунке 3.

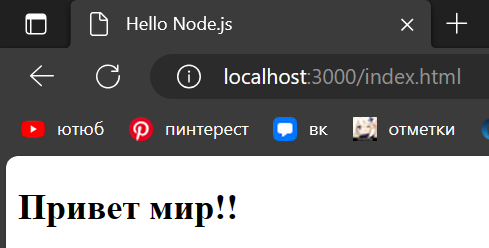


Рисунок 3 – Результат

**POST и GET запросы**

1. **GET запрос**

Листинг:

const http = require("http");

const url = require("url");

http.createServer(function (request, response) {

response.setHeader("Content-Type", "text/html; charset=utf8;");

const URL= url.parse(request.url, true)

console.log(URL.query)

response.end('ok')

}).listen(3000);

Результат работы представлен на рисунке 4, 5.

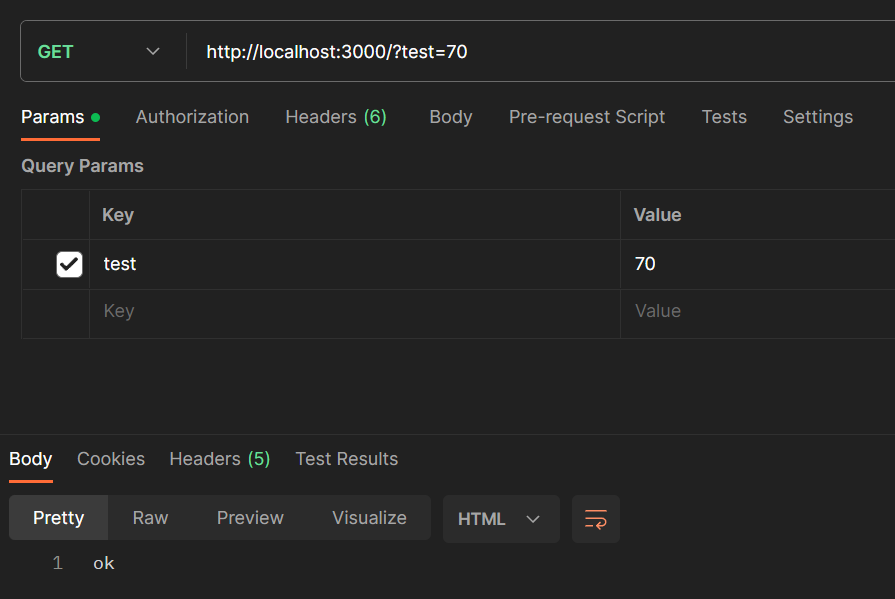


Рисунок 4 - Результат

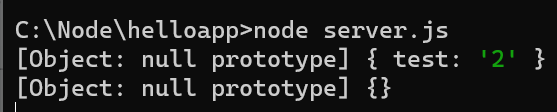


Рисунок 5 – Результат

1. **POST запрос**

Листинг:

const http = require("http");

const { parse } = require('querystring')

http.createServer(function (request, response) {

response.setHeader("Content-Type", "text/html; charset=utf8;");

let body = '';

request.on('data', chunk => {

body=chunk.toString()

console.log(body);

})

request.on('end', () => {

let params = parse(body)

console.log(params);

});

response.end("ok");

}).listen(3000);

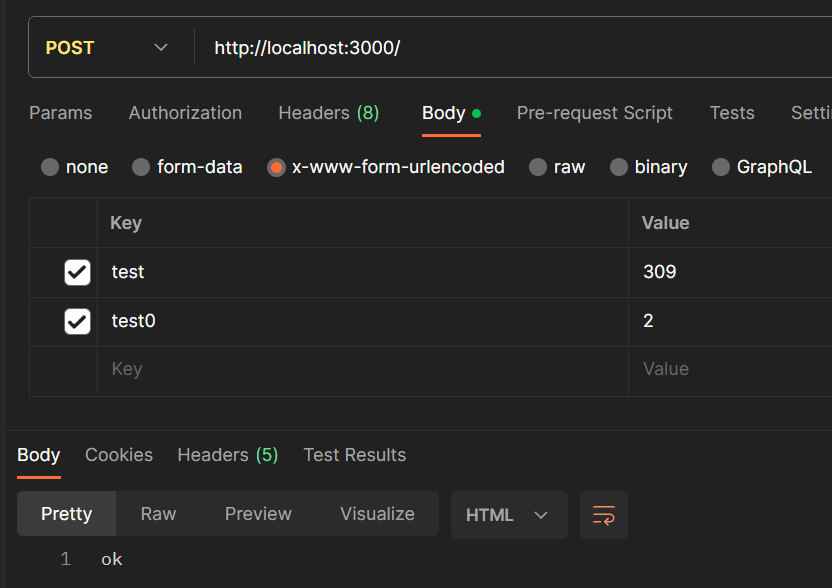


Рисунок 6 – Результат

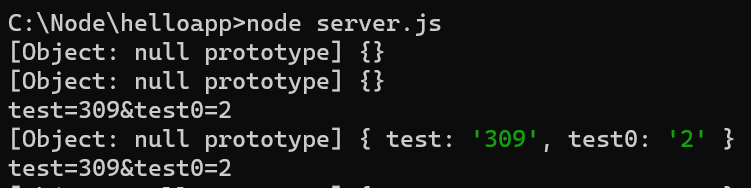


Рисунок 7 – Результат

**Маршрутизация**

Листинг:

const http = require("http");

http.createServer(function(request, response){

response.setHeader("Content-Type", "text/html; charset=utf8;");

if(request.url === "/home" || request.url === "/"){

response.write("<h2>Главная страница</h2>");

}

else if(request.url == "/about"){

response.write("<h2>О нас</h2>");

}

else if(request.url == "/contact"){

response.write("<h2>Контакты</h2>");

}

else{

response.write("<h2>Нет страницы</h2>");

}

response.end();

}).listen(3000);

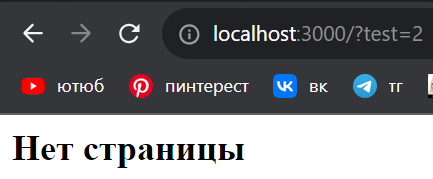


Рисунок 8 - Результат

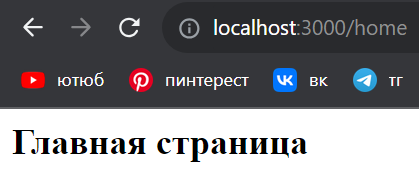


Рисунок 9 - Результат

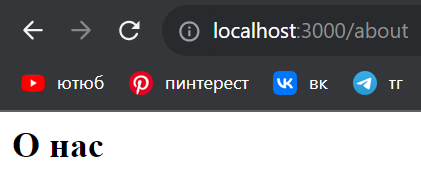


Рисунок 10 – Результат

**Переадресация**

Листинг:

const http = require("http");

http.createServer(function(request, response){

response.setHeader("Content-Type", "text/html; charset=utf8;");

if(request.url === "/"){

response.statusCode = 302; // временная переадресация

// на адрес localhost:3000/newpage

response.setHeader("Location", "/newpage");

}

else if(request.url == "/newpage"){

response.write("Новый адресс");

}

else{

response.statusCode = 404; // адрес не найден

response.write("Нет страницы");

}

response.end();

}).listen(3000);

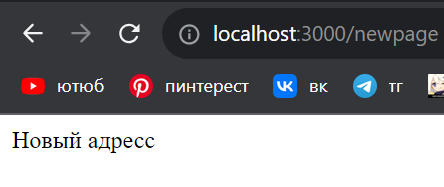


Рисунок 11 – Результат

**HTTPS-сервер**

Для создания сервера необходимо создать само подписанные сертификаты.

Необходимо создать файл закрытого ключа с помощью команды.

Листинг:

openssl genrsa -out ryans-key.pem 2048

2. Далее создаем файл Запроса на подписание сертификата с

помощью команды. Все запрашиваемые данные вводим.

Листинг:

openssl req -new -sha256 -key ryans-key.pem -out ryanscsr.pem

3. Создаем само подписанный сертификат с помощью команды.

Листинг:

openssl x509 -req -in ryans-csr.pem -signkey ryans-key.pem -out ryans-cert.pem

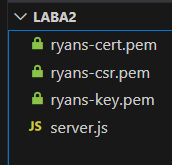


Рисунок 12 – самоподписанные сертификаты

Теперь самоподписанный сертификат можно подключить к https серверу.

Листинг:

const https = require('https');

const fs = require('fs');

const options = {

key: fs.readFileSync('./ryans-key.pem'),

cert: fs.readFileSync('./ryans-cert.pem'),

};

https.createServer(options, function (req, res) {

// код сервера

})

.listen(3000);

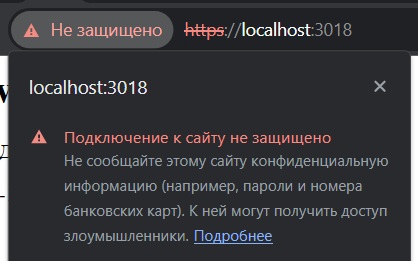
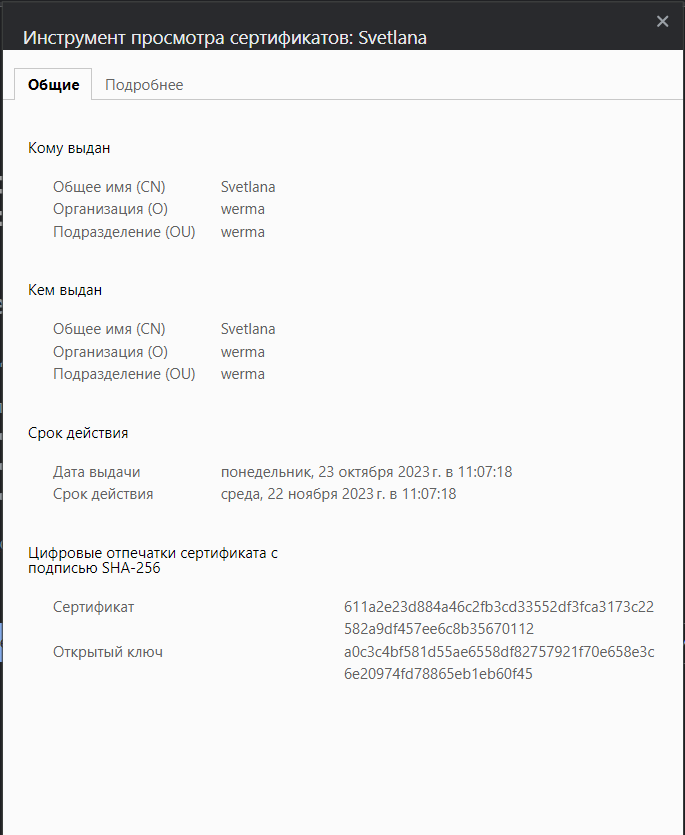


Рисунок 13 – подключение к https серверу



*Рисунок 14 - Сертификат*

Cоздан http-сервер, который решает следующие задачи:

- Работа с GET и POST параметрами

- Маршрутизация

- Переадресация

- Вывод на каждой странице вашего ФИО и группы.

Листинг:

const http = require("http");

const url = require("url");

const { parse } = require('querystring');

http.createServer(function (request, response) {

if (request.url === "/redirect") {

response.writeHead(302, {

'Location': '/home'

});

response.end();

}

const FIO = '<h2>ФИО: Орлова С.В.</h2>';

const group = '<h3>Группа: ИРсп-121</h3>';

response.setHeader("Content-Type", "text/html; charset=utf8;");

if (request.method === 'POST') {

let body = '';

request.on('data', chunk => {

body = chunk.toString()

})

request.on('end', () => {

let params = parse(body)

console.log(params);

});

}

if (request.method === 'GET') {

const URL = url.parse(request.url, true)

console.log(URL.query)

}

if(request.url === "/home" || request.url === "/"){

response.write(FIO);

response.write(group);

response.write("<h6>Главная страница</h6>")

}

else if(request.url == "/page"){

response.write(FIO);

response.write(group);

response.write("<h6>Вторая страница</h6>")

}

else{

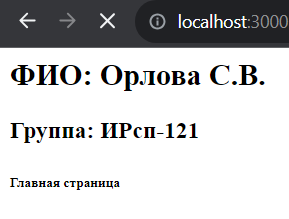
response.write("<h3>Такой страницы не существует</h3>")

}

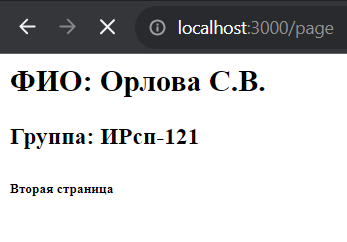
console.log('сервер запущен на порту 3000');

})

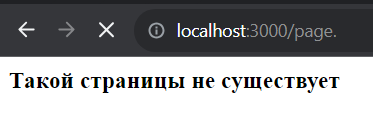
.listen(3000);



*Рисунок 15 - Результат*



*Рисунок 16 - Результат*



*Рисунок 17 - Результат*

создан https-сервер, который решает следующие задачи:

- Работа с GET и POST параметрами

- Маршрутизация

- Переадресация

- Вывод на каждой странице вашего ФИО и группы.

Листинг:

const https = require('https');

const fs = require('fs');

const { parse } = require('querystring');

const url = require('url');

const options = {

key: fs.readFileSync('./ryans-key.pem'),

cert: fs.readFileSync('./ryans-cert.pem'),

};

https.createServer(options, function (request, response) {

if (request.url === "/redirect") {

response.writeHead(302, {

'Location': '/home'

});

response.end();

}

const FIO = '<h2>ФИО: Орлова С.В.</h2>';

const group = '<h3>Группа: ИРсп-121</h3>';

response.setHeader("Content-Type", "text/html; charset=utf8;");

if (request.method === 'POST') {

let body = '';

request.on('data', chunk => {

body = chunk.toString()

})

request.on('end', () => {

let params = parse(body)

console.log(params);

});

}

if (request.method === 'GET') {

const URL = url.parse(request.url, true)

console.log(URL.query)

}

if (request.url === "/home" || request.url === "/") {

response.write(FIO);

response.write(group);

response.write("<h3>Главная страница</h3>")

}

else if (request.url == "/page") {

response.write(FIO);

response.write(group);

response.write('<h3>Вторая страница</h3>');

}

else {

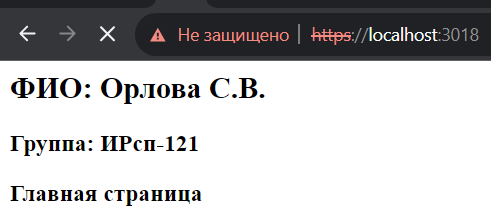
response.write("<h2>Такой страницы не существует</h2>")

}

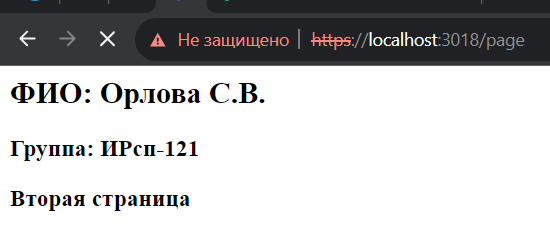
console.log('сервер запущен на порту 3018');

})

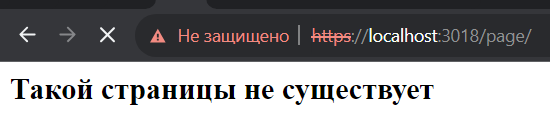
.listen(3018);



*Рисунок 18 - Результат*

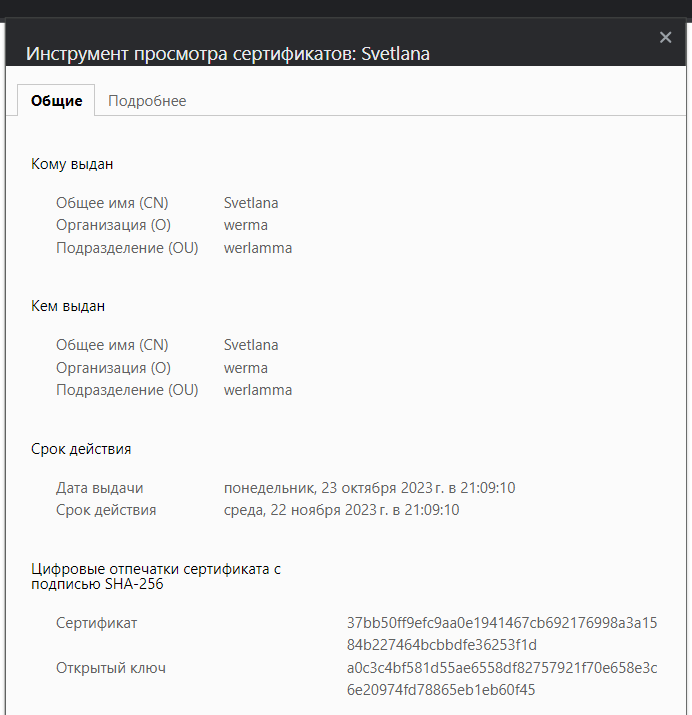


*Рисунок 19 - Результат*



*Рисунок 20 - Результат*

Самоподписанный сертификат:



*Рисунок 21 – Результат*

**Вывод**

В ходе данной лабораторной работы получены знания создания простейших HTTP серверов с помощью модуля http, получения GET и POST параметры и создания маршрутизации.

Так же получены знания создания самоподписанных HTTPS сертификатов, подключения HTTPS к HTTP-серверу NodeJS.